Package 'deco'

January 20, 2021

Type Package

Title Decomposing Heterogeneous Cohorts using Omic Data Profiling

Version 1.6.0

Date 2019-03-27

Author Francisco Jose Campos-Laborie, Jose Manuel Sanchez-Santos and Javier De Las Rivas. Bioinformatics and Functional Genomics Group. Cancer Research Center (CiC-IBMCC, CSIC/USAL). Salamanca. Spain.

Maintainer Francisco Jose Campos Laborie <fjcamlab@gmail.com>

biocViews Software, FeatureExtraction, Clustering, MultipleComparison, DifferentialExpression, Transcriptomics, BiomedicalInformatics, Proteomics, Bayesian, GeneExpression, Transcription, Sequencing, Microarray, ExonArray, RNASeq, MicroRNAArray, mRNAMicroarray

Description This package discovers differential features in hetero- and homogeneous omic data by a two-step method including subsampling LIMMA and NSCA. DECO reveals feature associations to hidden subclasses not exclusively related to higher deregulation levels.

Depends R (>= 3.5.0), AnnotationDbi, BiocParallel, SummarizedExperiment, limma

Imports stats, methods, ggplot2, foreign, graphics, BiocStyle, Biobase, cluster, gplots, RColorBrewer, locfit, made4, ade4, sfsmisc, scatterplot3d, gdata, grDevices, utils, reshape2, gridExtra

Suggests knitr, curatedTCGAData, MultiAssayExperiment, Homo.sapiens

License GPL (>=3)

URL https://github.com/fjcamlab/deco

ZipData true

LazyLoad true

VignetteBuilder knitr

RoxygenNote 5.0.1

NeedsCompilation no

git_url https://git.bioconductor.org/packages/deco

git_branch RELEASE_3_12

git_last_commit 910e12a

git_last_commit_date 2020-10-27

Date/Publication 2021-01-19

R topics documented:

---

deco-package *DECO: DEcomposing heterogeneous Cohorts from Omic profiling.*

Description

DECO integrates a two-step methodology to find out outlier behavior of features which could determine a new subclass of samples. Thus, a subsampling method with LIMMA (Stratified Differential Analysis or *RDA*) will be applied to a omic matrix data, previously normalized. After that, Non-Symmetrical Correspondence Analysis (*NSCA*) will be computed on frequency matrix of differential events generated by subsampling.

The feature-sample relationships will define stratified profiles with statistical significance. Any new subclass defined could be associated to hidden phenotypic variables.

Details

Package:	deco
Type:	Package
Version:	0.99
Date:	2018-11-15
License:	GPL 3.0

Author(s)

Francisco Jose Campos Laborie, Jose Manuel Sanchez Santos and Javier De las Rivas.

Maintainer: Francisco Jose Campos Laborie <fjcamlab@usal.es>, Cancer Research Centre (Salamanca) <jrivas@usal.es>

References

Campos-Laborie, FJ et al. **DECO: decompose heterogeneous population cohorts for patient stratification and discovery of sample biomarkers using omic data profiling.** (2018)

Scarfo, Irene et al. **Identification of a new subclass of ALK negative ALCL expressing aberrant levels of ERBB4 transcripts.** *Blood* (2015). http://dx.doi.org/10.1182/blood-2014-12-614503.

See Also

decoRDA, decoNSCA, decoReport, voom

Examples

```
#### ALCL EXAMPLE (Scarfo et al., 2015. Blood) ####
## Group-VS-group comparison

########################
# Loading example data #
########################
# Data from two subtypes (ALK+ and ALK-) of Anaplastic Large Cell Leukemia (ALCL).
data(ALCLdata)

## Classes vector to run a supervised analysis to compare both classes.
classes.ALCL <- colData(ALCL)[,"Alk.positivity"]
names(classes.ALCL) <- colnames(ALCL)

###################################
# Parallelization via BiocParallel #
###################################
# Non-parallel computing
bpparam <- SerialParam()

# Computing in shared memory
# all cores by deault
bpparam <- MulticoreParam()

#########################################################################
# RUNNING SUBSAMPLING OF DATA: BINARY design (two classes of samples) #
#########################################################################
# if annotation and rm.xy == TRUE, then
# library(Homo.sapiens)

# Not run as example
# sub.ma.3r.1K <- decoRDA(data = assay(ALCL), classes = classes.ALCL, q.val = 0.01,
#              rm.xy = TRUE, r = NULL, control = "pos", annot = FALSE, bpparam = bpparam,
#                id.type = "ENSEMBL", iterations = 10000, pack.db = "Homo.sapiens")


#############################################################################################
# RUNNING NSCA STEP: Looking for subclasses within a category/class of samples compared #
#############################################################################################
# Not run as example
# deco.results.ma <- decoNSCA(sub = sub.ma.3r.1K, v = 80, method = "ward.D", bpparam = bpparam,
#                         k.control = 3, k.case = 3, samp.perc = 0.05, rep.thr = 10)


# Phenotypical data from TCGA RNAseq samples.
```

```
colData(ALCL)

#########################################################
# PDF report with feature-sample patterns or subgroups #
#########################################################
## Generate PDF report with relevant information and several plots.

## Binary example (ALK+ vs ALK-) -not run as example-
# decoReport(deco.results.ma, sub.ma.3r.1K,
#           pdf.file = "report_example_microarray_binary.pdf",
#           info.sample = as.data.frame(colData(ALCL)[,8:10]),
#           cex.names = 0.3, print.annot = TRUE)
```

`ALCLdata` *Subset of microarray gene expression data from Anaplastic Large Cell Leukemia disease.*

Description

RMA normalized microarray gene expression data from clinical samples of Anaplastic Large Cell Leukemia (ALCL) patients. ALCL subtypes ALK+ (n=11) and ALK- (n=20). Corresponding to GSE65823 (GEO database ID) on Affymetrix HGU133Plus2.0 platform mapped to ENSEMBL genes with *genemapperhgu133plus2cdf* CDF package from **GATExplorer** website. RMA (Robust Multi-Array Average) normalization of microarrays and filtering of genes were applied (from 20172 genes to 1000 genes). Used as example of group-VS-group (binary) comparison with outlier behavior within ALK- samples.

Usage

```
data(ALCLdata)
```

Format

Formal class 'ExpressionSet' [package "Biobase"] with 7 slots.

Value

A matrix (1000 rows x 31 columns) containing the gene expression data (RMA normalized microarray data).

Source

GEO database ID: GSE65823

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE65823

GATExplorer CDF genemappers:

https://doi.org/10.1186/1471-2105-11-221

References

Scarfo, Irene et al. **Identification of a new subclass of ALK negative ALCL expressing aberrant levels of ERBB4 transcripts.** *Blood* (2015). http://dx.doi.org/10.1182/blood-2014-12-614503.

Examples

```
data(ALCLdata)

## list objects included
ls()
# [1] "ALCLdata"        "sub.ma.3r.1K"    "deco.results.ma"

# phenodata of each sample
head(colData(ALCL))

# classes vector
classes.ALCL <- colData(ALCL)[,"Alk.positivity"]
names(classes.ALCL) <- colnames(ALCL)

# gene expression matrix with 1000 ENSEMBL genes (mapped using GATExplorer genemappers)
# and 31 samples (11 ALK+ and 20 ALK-)
dim(assay(ALCL))
# [1] 1000    31

# density plot of RMA gene values from all dataset
plot(density(assay(ALCL)))
```

`AnnotateDECO` *DECO annotating function*

Description

This function allows to annotate different ID types using *Bioconductor* annotation packages (i.e. *Homo.sapiens, Mus.musculus, org.Hs.eg.db, org.MeSH.Spo.972h.db, org.Mm.eg.db ...*).

Usage

```
AnnotateDECO(ids, id.type, attributes = NA, pack.db = "Homo.sapiens",
                      rm.redundant = TRUE, verbose = FALSE)
```

Arguments

`ids`	a character vector including IDs to annotate.
`id.type`	a character indicating what type of ID we are managing. It should match with correct ID types including in annotation package ("Homo.sapiens" by default).
`attributes`	a character indicating which types of ID should annotate the original IDs. It should match with correct ID types including in annotation package (*Homo.sapiens* by default).
`pack.db`	name of the Bioconductor annotation package to use.
`rm.redundant`	logical indicating if redundant annotation (i.e. two different HGNC symbols for the same EntrezID) should be removed or not.
`verbose`	logical indicating if messages should be displayed.

Value

This function returns a `data.frame` containing original IDs and corresponding attributes to each one.

Author(s)

Francisco Jose Campos Laborie <fjcamlab@usal.es>

Examples

```
##### Annotation example
# Select an appropiate 'id.type' depending on which annotation
# package will be used.
# library(Homo.sapiens)
# columns(Homo.sapiens)

# Selecting first 10 original IDs from ALCL dataset
data(ALCLdata)
ids <- rownames(assay(ALCL))[1:10]

## Not run as example
# annot <- AnnotateDECO(ids, id.type = "ENSEMBL",
#                       attributes = c("SYMBOL","CHR","GENENAME"),
#                       pack.db = "Homo.sapiens", rm.redundant = TRUE)
```

`deco-class` *Class* `"deco"`

Description

A list R object returned by `decoNSCA` function from DECO R package.

Objects from the Class

Objects can be created by calls of the form `new("deco",...)`.

Slots

`featureTable`**:** Object of class `"data.frame"`: table of results with feature statistical information about NSCA merged with RDA information contained in 'subStatFeature' table from *decoRDA* function results.

`NSCAcluster`**:** Object of class `"list"`: object from *NSCAcluster* intern R function containing information about NSCA. If 'Binary' analysis was previously set, two lists corresponding to each class are contained (Control and Case).

`incidenceMatrix`**:** Object of class `"data.frame"`: absolute frequency matrix of 'm' features by 'n' samples size that summarizes differential events per feature per sample. It is essential for Non-symmetrical correspondence analysis.

`classes`**:** Object of class `"factor"`: vector with class labels.

`pos.iter`**:** Object of class `"numeric"`: number of iterations with DE signal.

`control`**:** Object of class `"character"`: control label.

`q.val`: Object of class `"numeric"`: adjusted.p.value threshold to LIMMA.

`rep.thr`: Object of class `"numeric"`: number of minimum repeats per sample to establish a differential event threshold.

`samp.perc`: Object of class `"numeric"`: minimum amount of samples showing `rep.thr` number of differential events to consider one feature as relevant (non-noisy) after RDA or subsampling step.

`subsampling.call`: Object of class `"numeric"`: number of minimum repeats per sample to establish a differential event threshold.

`nsca.call`: Object of class `"numeric"`: number of minimum repeats per sample to establish a differential event threshold.

Methods

summary `summary(object = "deco")`: Prints a summary of the information inside a 'deco' R object.

show `show(object = "deco")`: Prints the information inside a 'deco' R object.

featureTable `featureTable(object = "deco")`: Extract the featureTable containing all feature statistics.

NSCAcluster `NSCAcluster(object = "deco")`: Extract the NSCAcluster list containing all information and tables derived from decoNSCA step.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@usal.es>

Examples

```
showClass("deco")

showMethods("deco")
```

`decoNSCA` *Non-Symmetrical Correspondence Analysis (NSCA) applied to a subsampled dataset*

Description

This function needs R object from 'decoRDA' function to operate. It will applied NSCA to the frequency matrix of differential events, so it could calculate 'h' statistic per feature per samples to find out new subclasses of samples.

Usage

```
decoNSCA(sub, v = 80, k.control = NULL,
         k.case = NULL, rep.thr = 3,
         bpparam = SerialParam(),
         samp.perc = 0.05,
         method = "ward.D")
```

Arguments

sub	an output R object from 'decoRDA' R function.
v	minimum percentage of variability that NSCA have to explain.
k.control	number of subclasses to find out within 'Control' samples (binary contrast) or all samples (unsupervised contrast).
k.case	number of subclasses to find out within 'Case' samples (binary contrast).
rep.thr	numeric, it should correspond to the minimum number of differential events that should have each sample within 'samp.perc' percentage at least. Both 'rep.thr' and 'samp.perc' compounds a threshold to remove noisy features not explicity associated to any subgroup of samples.
bpparam	registered parallelization using BiocParallel R functions: SerialParam(), MulticoreParam(), SnowParam(), etc. Further information available in vignette.
samp.perc	numeric, it should correspond to the minimum percentage of samples having to be affected with a minimum 'rep.thr' number of differential events per feature. Both 'rep.thr' and 'samp.perc' compounds a threshold to remove noisy features not explicity associated to any subgroup of samples.
method	character indicating which agglomerative method should be used to generate sample dendrogram. If is NULL (by default), all possible methods would be tested and corresponding to the highest cophenetic correlation would be selected.

Details

Once we obtain frequency matrix of DE events or *incidenceMatrix*, we applied a **NSCA** procedure. NSCA let us to analyse all dependendent structures between differential features and samples derived from the same relational space (dataset).

In this context, NSCA stablishes an assymetric and directional association between features (response) and samples (predictor), so each sample contributes in a particular way to the differential expression of a singular feature. Thus, an outlier subsets of samples for a feature could explain most of its differential expression signal, allowing us to assign this feature with these samples (see figure of *incidenceMatrix* above). Further information about how NSCA functions and other properties are detailed by Lombardo et al.

Value

Returns a "deco" object with following slots:

featureTable	table of results with gene or feature statistical information about NSCA merged with subsampling information contained in previous 'diffgenes' value from 'decoRDA' function results.
NSCAcluster	list from 'NSCAcluster' R function containing information about NSCA. If 'Binary' analysis was previously set, two lists corresponding to each class are contained.
incidenceMatrix	absolute frequency matrix of 'd' genes by 's' samples size that summarizes DE events for each gene per sample. It will be essential for Non-symmetrical correspondence analysis.
classes	factor or character vector of 's' size provided by user.
control	label if control class was defined.

`pos.iter`	lowest number of repeats after applying 'decoNSCA' threshold.
`q.val`	adjusted.p.value threshold defined to each subsampling iteration with LIMMA within 'decoRDA' function.
`subsampling.call`	call to decoRDA previously run.
`deco.call`	call to decoNSCA previously run.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@gmail.com>

References

Lauro, N. and D'Ambra, L. (1984). **L'analyse non symetrique des correspondances.** *Data Analysis and Informatics*

Beh, E.J. and Lombardo, R. (2014). **Correspondence Analysis. Theory, Practice and New Strategies**. *John Wiley & Sons*

See Also

`decoRDA`, `hclust`

Examples

```
## User has to provide a RDA R object, running decoRDA() previously
data(ALCLdata)

## Slots included in the returned RDA object are:
names(sub.ma.3r.1K)
# [1] "data"            "results"         "subStatFeature"  "incidenceMatrix" "classes"
# [6] "resampleSize"    "control"         "pos.iter"        "q.val"           "call"

# Computing in shared memory
# all cores by deault
bpparam <- MulticoreParam()

##############################################################################################
# RUNNING NSCA STEP: Looking for subclasses within a category/class of samples compared #
##############################################################################################

# Not run as example
# deco.results.ma <- decoNSCA(sub = sub.ma.3r.1K, v = 80,
#         method = "ward.D", bpparam = bpparam,
#         k.control = 3, k.case = 3, samp.perc = 0.05, rep.thr = 5)


## Class 'deco'
class(deco.results.ma)
# [1] "deco"


## Slots included within 'deco' R object

# slotNames(deco.results.ma)
# [1] "featureTable"    "NSCAcluster"     "incidenceMatrix"  "classes"        "pos.iter"
```

```
# [6] "control"       "q.val"         "rep.thr"       "samp.perc"      "subsampling.call"
#[11] "nsca.call"


## Top-10 features from DECO analysis based on "Standard.Chi.Square"
head(featureTable(deco.results.ma)[order(featureTable(deco.results.ma)$Standard.Chi.Square,
     decreasing = TRUE),], 10)


## Matrix of 'h statistic' calculated by DECO per category of samples
# if binary analysis was carried out.
dim(NSCAcluster(deco.results.ma)$Control$NSCA$h)
dim(NSCAcluster(deco.results.ma)$Case$NSCA$h)


## Top-10 discriminant features for CASE samples (ALK-) based on
# 'h statistic per subclass found.
head(NSCAcluster(deco.results.ma)$Case$rankingFeature.h, 10)


### Sample and subclass information could be found in two slots within 'NSCAcluster':
## General information about subclasses
# NSCAcluster(deco.results.ma)$Case$infoSubclass

## Sample membership to a subclass.
# NSCAcluster(deco.results.ma)$Case$samplesSubclass


## Both 'hclust' dendrogram information of CASE samples and features.
## They include Huber's Gamma coefficient value and Cophenetic
## correlation between dendrogram distances
# and distance matrix.

#names(NSCAcluster(deco.results.ma)$Case$hclustSamp)
#[1] "dend"    "coph"    "cluster" "huber"

#names(NSCAcluster(deco.results.ma)$Case$hclustFeat)
#[1] "dend"    "coph"    "cluster" "huber"

###### Get summary info about "DECO" analysis:
###
summary(deco.results.ma)
# Decomposing Heterogeneous Cohorts from Omic profiling: DECO
# Summary:

# Analysis design: Binary
# Classes compared:
# neg pos
#  20  11

#            RDA.q.value Minimum.repeats Percentage.of.affected.samples NSCA.variability
# Thresholds        0.01           10.00                           5.00             84.1

# Number of features out of thresholds: 255
# Feature profile table:
# Complete Majority Minority
#       12       79      164
```

```
# Number of samples affected: 31
# Number of positive RDA comparisons: 1955
# Number of total RDA comparisons: 10000

###### Get info about "DECO" analysis
show(deco.results.ma)

#########################################################################
# RUNNING NSCA STEP: Looking for subclasses within all samples compared #
#########################################################################

## Not run as example
# deco.results.ma.uns <- decoNSCA(sub = sub.ma.3r.1K.uns, v = 80, method = "ward.D",
#                         k.control = 3, k.case = 3, samp.perc = 0.05, rep.thr = 3)

########################################
# RUNNING NSCA STEP: Multiclass design #
########################################

## Not run as example
# deco.results.ma.multi <- decoNSCA(sub = sub.ma.3r.1K.multi, v = 80, method = "ward.D",
#                         k.control = 3, k.case = 3, samp.perc = 0.05, rep.thr = 3)
```

`decoRDA` *Subsampling function to find out differential events along samples from an omic dataset*

Description

RDA R function within 'deco' R package. This function allows user to subsampling a omic data matrix to find out significant differential features characterizing any hidden subclass or group of samples included.

Usage

```
decoRDA(data, classes = NA, control = NA, r = NULL, q.val = 0.01,
              iterations = 10000, bpparam = SerialParam(), annot = FALSE,
              id.type = NA, attributes = NA, rm.xy = FALSE,
              pack.db = "Homo.sapiens")
```

Arguments

`data`	matrix of normalized omic data with 'f' features (rows) by 'n' samples (columns).
`classes`	factor or character vector of 's' size with two ('Supervised' or 'Binary' analysis) or more classes ('Multiclass' analysis) to contrast. Names of vector must match with colnames of 'data'. If any vector of classes is given, 'Unsupervised' analysis will be run with all samples.
`control`	character label defining 'control' or '0' class for LIMMA contrast design.
`r`	number of samples of each class to take for subsampling. Same number of samples will be taken from each class.

q.val	p.value threshold for each iteration of subsampling process. It corresponds to *adj.P.value* of LIMMA and is set as default as 0.01.
iterations	number of iterations to subsample 'data', set as default as 10000. If this value exceeds number of all possible combinations, it would be replaced.
bpparam	registered parallelization using BiocParallel R functions: SerialParam(), MulticoreParam(), SnowParam(), etc. Further information available in vignette.
annot	logical value indicating if annotation of IDs provided by 'data' as rownames should be done.
id.type	character indicating what ID is provided by user in 'data'. This value must match any of distinct possibles values of R package used to annotate.
attributes	character vector with distinct fields to return by annotation package.
rm.xy	logical indicating if features located on chromosomes 'X' or 'Y' have to be removed. Features depending on these location could be an artefact of unbalanced contrasts by gender of patients or samples, so removing it should clear final results due to an appropiated subsampling of samples.
pack.db	character naming annotation package to be used. *Homo.sapiens* annotation package set as default, try first 'AnnotateDECO' R function with any other annotation package from Bioconductor to find out any problem.

Details

The **RDA step** is primarly conditioned by contrast design. Our capacity to highlight *majority* or *minority* features will vary depending on how much we want focusing the analyisis on classes or individual samples, or what is the same, the **granularity of RDA**.

In this way, it is necessary to highlight that two **RDA strategies** could be: (a) a **majority** one trying to improve our generic differential expression with higher subsampling size; and (b) a **minority** strategy to find out all possible hidden subclasses. By default, `decoRDA()` defines an optimal subsample size *r*. Let *n* be the number of samples included in the analysis or

$$n_1, n_2, ..., n_k$$

in case of two or more *k* classes, then based on previous analysis done by Babu et al.:

$$r = \sqrt{n}$$

or in case of 'Supervised' or 'Multiclass' design:

$$r = \sqrt{min(n_1, n_2, ..., n_k)}$$

The analysis can be set to find differences among two populations or classes and to find differences within the whole cohort of samples. Besides, `decoRDA()` can work under both scenarios: **binary** analysis with two classes or **unsupervised** analysis contrasting all samples. The *classes* input is used to define it. If the user introduces any vector with labels of two classes for each sample, a **binary** analysis will be run. Otherwise, without any *classes* vector all samples will be compared with each other.

Value

Returns a list containing:

data	input matrix of normalized data with 'f' features (rows) by 's' samples (columns).

subStatFeature	table of differential features with statistical and annotation information, if it was required.
incidenceMatrix	absolute frequency matrix of 'd' features by 's' samples size that summarizes differential events for each feature per sample. It will be essential for Non-Symmetrical Correspondence Analysis.
classes	factor or character vector of 's' size provided by user.
resampleSize	number of samples of each class to take for subsampling provided by user in 'r'.
control	character label defining 'control' or '0' class for LIMMA contrast design.
pos.iter	number of subsampling iterations showing at least 1 differential event.
q.val	*adj.p.value* threshold used in each LIMMA iteration within subsampling.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@usal.es>

See Also

decoNSCA, voom

Examples

```
#### ALCL EXAMPLE (Scarfo et al., 2015. Blood) ####

########################
# Loading example data #
########################
## Data from two subtypes (ALK+ and ALK-) of Anaplastic Large Cell Leukemia (ALCL).
data(ALCLdata)

## Classes vector to run a binary analysis to compare both classes.
classes.ALCL <- colData(ALCL)[,"Alk.positivity"]
names(classes.ALCL) <- colnames(ALCL)

#########################################################################
# RUNNING SUBSAMPLING OF DATA: BINARY design (two classes of samples) #
#########################################################################
# library(Homo.sapiens) # for gene annotation

## Not run as example
# sub.ma.3r.1K <- decoRDA(data = assay(ALCL), classes = classes.ALCL, q.val = 0.01,
#               rm.xy = TRUE, r = NULL, control = "pos", annot = TRUE,
#               id.type = "ENSEMBL", iterations = 1000,
#               pack.db = "Homo.sapiens")


## Slots included in returned R object.

# names(sub.ma.3r.1K)
# [1] "data"           "results"        "subStatFeature"  "incidenceMatrix" "classes"
# [6] "resampleSize"   "control"        "pos.iter"        "q.val"           "call"


## Top-10 RDA features.
```

```
# head(sub.ma.3r.1K$subStatFeature, 10)


#####################################################################
# RUNNING SUBSAMPLING OF DATA: UNSUPERVISED design (no classes) #
#####################################################################

## Not run as example
# sub.ma.3r.1K.uns <- decoRDA(data = assay(ALCL), q.val = 0.01,
#                       rm.xy = TRUE, r = 3, annot = TRUE,
#                       id.type = "ENSEMBL", iterations = 1000,
#                       pack.db = "Homo.sapiens")



#####################################################################
# RUNNING SUBSAMPLING OF DATA: MULTICLASS design (no classes) #
#####################################################################

# 3 classes: ALK+, PTCL and ALK- without PTCLs
multiclasses.ALCL <- factor(apply(
    as.data.frame(colData(ALCL)[, c("Alk.positivity", "Type")]), 1,
    function(x) paste(x, collapse = ".")
))
head(multiclasses.ALCL)

## Not run as example
# sub.ma.3r.1K.multi <- decoRDA(data = assay(ALCL), classes = multiclasses.ALCL,
#                       q.val = 0.01, rm.xy = TRUE, r = 3, annot = TRUE,
#                       id.type = "ENSEMBL", iterations = 1000,
#                       pack.db = "Homo.sapiens")
```

`decoReport` *Report of DECO analyisis*

Description

This report includes a PDF file with several different representations from the analysis. Additionally, a text file (tabulated-delimited) including *featureTable* slot from a **deco** R object and a text file with ***h*** statistics per feature and subclass found were both generated. They will be created inside current working directory.

Usage

```
decoReport(deco, sub, id = NA, pdf.file = "decoReport.pdf",
                 info.sample = NA, info.feature = NA,
                 cex.samples = 1.2, cex.legend = 0.9,
                 cex.names = 0.8, print.annot = FALSE)
```

Arguments

deco	a "deco" R object generated by 'decoNSCA' R function.
sub	a R object generated by 'decoRDA' R function.

id	character vector indicating what original IDs from input data should be highlighted and plot in 'Profile' section. If all IDs found have to be plot, input all corresponding IDs here.
pdf.file	path or name of pdf where report should be generated.
info.sample	a factor or data.frame with relevant information of samples to be plot on 'Heatmap'.
info.feature	a factor or data.frame with relevant information of genes or features to be plot on 'Heatmap'.
cex.samples	numerical value giving the amount of magnification of sample related information, i.e. sample points in 'Profile' plots.
cex.legend	numerical value giving the amount of magnification of legend.
cex.names	numerical value giving the amount of magnification to gene names or sample names.
print.annot	logical indicating if annotation (gene SYMBOL) should be printed instead original IDs. This option needs previous annotation within 'decoRDA' function or late annotation binding information to @featureTable slot of 'deco' R object.

Value

Returns a 'pdf' file including relevant information about all DECO algorithm analysis.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@usal.es>

See Also

Vignette of R package explains all different plots included within the pdf.

decoRDA, decoNSCA

Examples

```
# Phenotypical data from TCGA RNAseq samples.
data(ALCLdata)
head(colData(ALCL))

#####################################################
# PDF report with gene-sample patterns or subgroups #
#####################################################
## Generate PDF report with relevant information and several plots.
# It will also generate a txt (tab-delimitated) table including featureTable.
# Both objects will be saved in your working directory if any path is given in PDF name.

## Binary example (ALK+ vs ALK-)
# decoReport(deco.results.ma, sub.ma.3r.1K,
#            pdf.file = "report_example_microarray_binary.pdf",
#            info.sample = as.data.frame(colData(ALCL))[,8:10],
#            cex.names = 0.3, print.annot = TRUE)

## Unsupervised example (no classes)
# decoReport(deco.results.ma.uns, sub.ma.3r.1K.uns,
#            pdf.file = "report_example_microarray_unsupervised.pdf",
#            info.sample = as.data.frame(colData(ALCL))[,8:10],
#            cex.names = 0.3, print.annot = TRUE)
```

```
## Multiclass example (ALK+ vs PTCL vs ALK-(noPTCL))
# decoReport(deco.results.ma.multi, sub.ma.3r.1K.multi,
#            pdf.file = "report_example_microarray_multi.pdf",
#            info.sample = as.data.frame(colData(ALCL))[,8:10],
#            cex.names = 0.3, print.annot = TRUE)
```

`featureTable` *Accessor to 'featureTable' slot of 'deco-class' objects.*

Description

The 'featureTable' slot is the main output table, including all the feature statistics and rankings from both steps of DECO: RDA and NSCA.

Usage

```
featureTable(object)
```

Arguments

`object`	a deco-class object.

Value

This function returns the featureTable within a deco-class object.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@gmail.com>

See Also

`decoRDA`

Examples

```
## Loading pre-calculated objects...
data(ALCLdata)

featTable <- featureTable(deco.results.ma)

head(featTable)
```

NSCAcluster *Accessor to 'NSCAcluster' slot of 'deco-class' objects.*

Description

The 'NSCAcluster' slot is the main output from NSCA step. I would be divided in 'Control' and 'Case' if only two categories of samples were input to first 'decoRDA()' function, or would include only one 'All' slot for multiclass or unsupervised analysis.

Usage

```
NSCAcluster(object)
```

Arguments

object	a deco-class object.

Value

This function returns the NSCAcluster list within a deco-class object.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@gmail.com>

See Also

decoRDA

Examples

```
##
## Loading pre-calculated objects...
data(ALCLdata)

resNSCA <- NSCAcluster(deco.results.ma)

head(resNSCA)

# Information within the 'case' category
str(resNSCA$Case)
```

`plotAssociationH` *Plot to visualize association among DECO subclasses and sample information (phenotype).*

Description

This function returns three plots showing the association among different sample subclasses found by DECO and any sample information of interest. This information can be the initial classes or any new included sample information.

Usage

```
plotAssociationH(deco, info.sample)
```

Arguments

`deco`	a "deco" R object generated by 'decoNSCA' R function.
`info.sample`	a character vector or factor including the sample information.

Details

This function provides a simple comparison of any phenotype characteristic and new DECO subclasses.

Value

Violin plot: distribution of average h-statistic values (per feature) within the newly provided categories of samples, splitting by the subsets of features associated to each DECO subclass.

Left-heatmap: frequency table of *samples* among DECO subclasses and newly provided sample categories. Right-heatmap: frequency table of *features* among DECO subclasses and newly provided sample categories.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@gmail.com>

See Also

`decoRDA`, `decoNSCA`

Examples

```
#### Further information in DECO vignette(s).
## Run after decoRDA and decoNSCA steps on 'ALCLdata'
data(ALCLdata)
ALCL

### Sample source information
info.sample <- as.data.frame(colData(ALCL))[,"Sample.origin"]
names(info.sample) <- rownames(colData(ALCL))

#### Comparing DECO subclasses against source of samples.
plotAssociationH(deco.results.ma, info.sample)
```

plotDECOProfile *Feature profile from DECO analysis*

Description

Plot feature profile(s) in a separate PDF from decoReport R function.

Usage

```
plotDECOProfile(deco, id, data, pdf.file = NA, plot.h = FALSE,
                      info.sample = NA, print.annot = FALSE,
                      cex.legend = 1.1, cex.names = 1, cex.samples = 1)
```

Arguments

deco	a "deco" R object generated by 'decoNSCA' R function.
id	character vector indicating what original IDs from input data should be highlighted and plot in 'Profile' section. If all IDs found have to be plot, input all corresponding IDs here.
data	input matrix of normalized data with 'f' features (rows) by 's' samples (columns).
pdf.file	path or name of pdf where report should be generated.
plot.h	logical indicating if points corresponding to 'h' relative values should be plotted.
info.sample	a factor or data.frame with relevant information of samples to be plot on 'Heatmap'.
print.annot	logical indicating if annotation (gene SYMBOL) should be printed instead original IDs. This option needs previous annotation within 'decoRDA' function or late annotation binding information to @featureTable slot of 'deco' R object.
cex.legend	numerical value giving the amount of magnification of legend.
cex.names	numerical value giving the amount of magnification to gene names or sample names.
cex.samples	numerical value giving the amount of magnification of sample related information, i.e. sample points in 'Profile' plots.

Value

Returns a new PDF.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@usal.es>

See Also

decoNSCA, voom

Examples

```
#### Further information in DECO vignette(s).
## Run after decoRDA and decoNSCA steps on 'ALCLdata'
data(ALCLdata)
ALCL

### ERBB4 gene profile
# plotDECOProfile(deco = deco.results.ma, id = "ENSG00000178568",
#                 data = assay(ALCL), cex.samples = 2,
#                 pdf.file = "ERBB4_profile_ALCL.pdf",
#                 info.sample = as.data.frame(colData(ALCL))[,c(9,8,10)])
```

`plotGainingH` *Gaining plots for using h-statistic instead original omic data.*

Description

This function returns three plots for a particular feature(s), showing the correlation between the h-statistic calculated by DECO and the original omic data.

Usage

```
plotGainingH(deco, data, ids, print.annot = FALSE,
             orig.classes = TRUE)
```

Arguments

`deco`	a "deco" R object generated by 'decoNSCA' R function.
`data`	a matrix of normalized omic data with 'f' features (rows) by 'n' samples (columns).
`ids`	character vector indicating what original IDs from input data should be highlighted and plot in 'Profile' section. If all IDs found have to be plot, input all corresponding IDs here.
`print.annot`	logical indicating if annotation (gene SYMBOL) should be printed instead original IDs. This option needs previous annotation within 'decoRDA' function or late annotation binding information to @featureTable slot of 'deco' R object.
`orig.classes`	logical indicating if original sample categories compared (orig.classes = TRUE) or DECO subclasses (orig.classes = FALSE) will be represented.

Details

Diagnostic plot associating omic profile and h-statistic profile of any significant feature found by DECO.

Value

Three plots are returned: Boxplot: distribution of omic data and h-statistic values per DECO subclass. Top-left: parametric correlation between omic data and h-statistic per sample. Top-right: non-parametric correlation (ranking) between omic data and h-statistic per sample.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@gmail.com>

See Also

decoRDA, decoNSCA

Examples

```
#### Further information in DECO vignette(s).
## Run after decoRDA and decoNSCA steps on 'ALCLdata'
data(ALCLdata)
ALCL

## Feature to represent
id <- featureTable(deco.results.ma)[1, "ID"]

#### Comparing DECO subclasses against source of samples.
plotGainingH(deco.results.ma, data = assay(ALCL), ids = id,
             print.annot = FALSE, orig.classes = FALSE)
```

`plotHeatmapH` *Heatmap corresponding to h-statistic matrix provided by DECO analysis.*

Description

This function returns three plots showing the association among different sample subclasses found by DECO and any sample information of interest. This information can be the initial classes or any new included sample information.

Usage

```
plotHeatmapH(deco, info.sample = NA, info.feature = NA,
      print.annot = FALSE, cex.legend = 1,
      cex.names = 1)
```

Arguments

`deco`	a "deco" R object generated by 'decoNSCA' R function.
`info.sample`	a data.frame with relevant information of samples to be plot on 'Heatmap'.
`info.feature`	a data.frame with relevant information of genes or features to be plot on 'Heatmap'.
`print.annot`	logical indicating if annotation (gene SYMBOL) should be printed instead original IDs. This option needs previous annotation within 'decoRDA' function or late annotation binding information to @featureTable slot of 'deco' R object.
`cex.legend`	numerical value giving the amount of magnification of legend.
`cex.names`	numerical value giving the amount of magnification to gene names or sample names.

Details

Heatmap plot showing h-statistic matrix (H) and biclustering of features-samples.

Value

A customized heatmap plot showing h-statistic matrix and biclustering of features-samples.

Author(s)

Francisco Jose Campos Laborie. <fjcamlab@gmail.com>

See Also

decoReport, decoNSCA

Examples

```
#### Further information in DECO vignette(s).
## Run after decoRDA and decoNSCA steps on 'ALCLdata'
data(ALCLdata)
ALCL

### Phenotype information
info.sample <- as.data.frame(colData(ALCL))[,8:9]
rownames(info.sample) <- rownames(colData(ALCL))

#### Heatmap with h-statistic matrix and biclustering of features-samples.
## Not run as example
# plotHeatmapH(deco = deco.results.ma, info.sample = info.sample,
#              cex.names = 0.3, print.annot = FALSE)
```

Index